Tessellating and Searching Uncertain State Spaces

A. Meystel, A. Bathija

Department of Electrical and Computer Engineering, Drexel University
Philadelphia, PA 19104

Abstract
Multiresolutional S^3-search generated a need to properly tessellate spaces and efficiently searching them. Drexel University has introduced MR-methodology such as uniform and non-uniform space tessellation and efficient algorithms of searching within ressellated state space. This methodology for solving planning and control problems is successfully applied in autonomous vehicles, industrial robots and power stations. This paper focuses on computational phenomena characteristic for randomized tessellation and affecting the results of S^3-search.

Keywords: bias, Dijkstra algorithm, envelope, multiresolutional, randomized tessellation, shaking the grid, search in the state space, stripe, uniform and non-uniform tessellation, variable traversability

Introduction

To "tessellate" to form or arrange elementary units of space in a mosaic fashion so that all selected space be covered. Tessellation of a closed polytope means decomposing it into non-overlapping sets of tessellata (granules, boxes or tiles). These subsets form an equivalence class in which all the points within a tile are identified with a single label. The natural interpretation is that each of the labels represents all points of a single tessellatum while its generalized location is in the center of the tessellatum. There are three basic types of tessellations:

1. Regular Tessellation
2. Semi Regular Tessellation
3. Irregular Tessellation

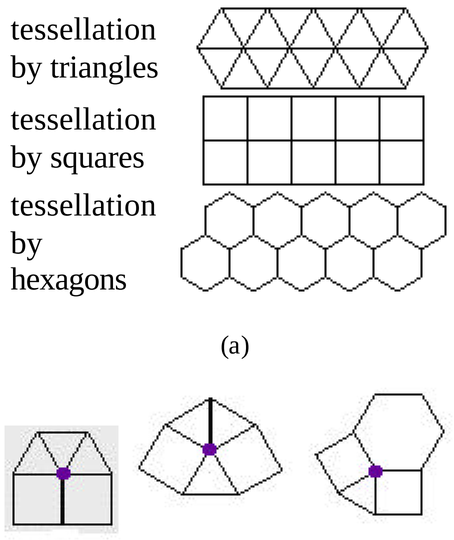
Regular Tessellation: This means a tessellation made up of congruent regular polygons. *Regular* means that the sides of the polygon are all the same length. *Congruent* means that the polygons that you put together are all the same size and shape. Only three regular polygons tessellate in the Euclidean plane: triangles, squares or hexagons. Regular tessellation are shown in figure 1.a.
Semi-Regular Tessellation: A variety of regular polygons is used to make **semi-regular tessellations**. Examples of this type of tessellation are shown in Figure 1.b. This tessellation has two properties:

1. It is formed by regular polygons.
2. The arrangement of polygons at every vertex point is identical.

Irregular Tessellation: In this case points are sprinkled randomly. This randomity is equivalent to the uncertainties o sampling the state space. There are quite a few methods of generating random tessellations.

Generating randomized tessellation.

S^3-search requires generating random points in the state space [1]. The law of distribution that characterizes coordinates of the random points describes the uncertainty of the state space taken into consideration. There are many methods of generating randomized graphs (Irregular tessellation). A few of them are listed below.



(b)
Figure 1: Different types of tessellation

1. ***Random generation of grid-points positions*:** Random points in the state space can be generated by a standard random number generator. A deficiency of this method is in the fact that "random" points may not be evenly distributed. In other words, the density of the particular set of points in all regions of the state

space may not be the same. This may occur because of dealing with imperfect random number generators. However, this shortcoming can be dealt with by multiple running the algorithm that is being tested.

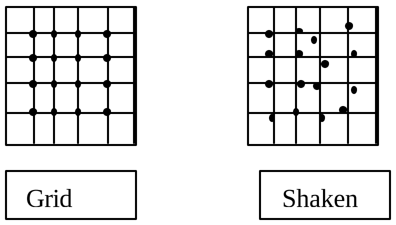


Figure 2: Shaken grid method of generating random points

2. ***Randomizing by "shaking the grid":*** In this method we use a grid to generate the basic set of points. The grid points are considered to be at the intersection of a row and a column. Then we introduce a random shift to these coordinates of points: shake the grid. Thus, the points move away from the initial positions into random positions.

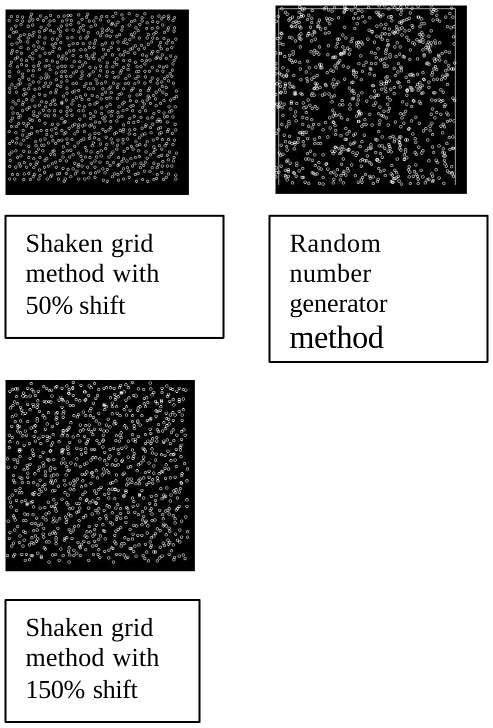


Figure 3: Comparison of the various methods for generating the grid

The shift is assigned in the form of a fraction (percentage) of the interval between the rows and the columns (Δ). Figure 2 explains the generation of the randomized tessellation using the grid method. The advantage of this method is that it ensures that the *overall density* of the points is the same in all the regions of the graph. But this true only if the shift in the points is less than or equal to 50% (.5Δ). If the shift exceeds 50% the distribution gets similar to the one observed in the random number generation method. This comparison is shown in Figure 3.

3. ***Concentric circle non-uniform grid generation:*** This method is similar to the *grid generation method.* In the latter we used straight lines to generate the grid. But in this case we use concentric circles instead of the rows and the radius replaces columns. The specifics of this method is that there are many points generated close to the center of the circle and as we move away from the center the density of the points decrease. This method is useful in the cases when a variable resolution is attempted within one level of representation like in page 4 of [2], where the fine motion planning is obtained for a few steps in the vicinity and the further we move from the acting robot the lower the resolution is. Certainly, any law of gradual decrease of resolution can be assigned. In Figure 4: we illustrate how the random points are generated by this method.

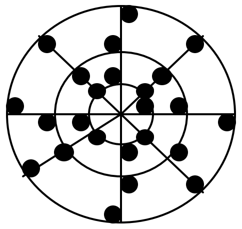


Figure 4: Non-uniform Random points generation using concentric circles

Randomized Grid Generation

The algorithm of randomized grid generation is described by this pseudocode:

```
1. Start.
2. Obtain the parameter of the grid
    a. Total number of points
    b. Total number of rows
    c. Total number of columns
    d. Check if the product of the rows and the columns is equal to the total number of points. It true then continue else go to step 2.a
    e. Obtain the starting coordinate of the row and the column.
    f. Obtain the delta for the row. Check if the obtained value is valid.
```

```
        g.  Obtain the delta for the column. Check
            if the entered value is valid
        h.  Obtain the shift percentage of the
            points.
        i.  Calculate the max possible shift of a
            point with respect to the row and the
            column delta.
3.  Repeat the following for total number of
    rows (columns).
            i.   Assign a random sign (+ or -)
            ii.  Obtain a random value with the
                 max possible shifts for each of the
                 row and column.
            iii. Add this shift to the corresponding
                 row and column coordinate.
        4.  Store this information
5.  Stop.
```

Randomized graph using different distributions.

In the above pseudocode, we assign the "sign" that a random generator uses to obtain the random shift within the maximum limits. Different distributions can be used to obtain this random value. Uniform distribution is the distribution that we have used in our experiments. In uniform distribution the random point obtained has equal probability of falling anywhere in the space specified for it. While if we use the Guassian Distribution then the points have a greater probability of falling close to the center of the shift compared to the probability of falling at the edge of the maximum shifts.

Searching on the Grid Problem: Why Bias?

This is a useful advantage and the disadvantage of a grid that from any node in the grid to another there are many equal (or approximately equal) paths. This can be seen in Figure 5.

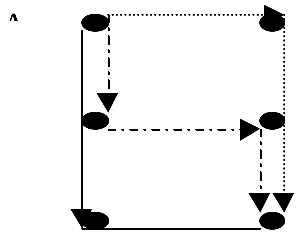


Figure 5: Three possible paths from node A to node B in the grid

In the above Figure, it is clear that there are 3 possible equivalent paths from node A to node B. On using the new algorithm on the grid, it tends to give the full line path (in Figure 5) as the optimal solution. It turns out to be that if the starting node is on the left hand side and the destination point is on the right hand side and there are many possible paths from the starting to the destination then the left most path will be given as the optimal solution. Similarly if the starting point in the right hand side and the destination on the left hand side, then the right most path among the possible paths will be given as the optimal one. For a single run of the min-cost search see Figure 6.

The concept of Vicinity

Unlike the standard Dijkstra search algorithm, our search in the randomized grid does not have any graph prepared for exploration in advance for exploration. The recommended algorithms builds the graph as it explores the graph. The concept of a vicinity is introduced. The present "standpoint" is being surrounded by a "vicinity" that can have a "radius" of 1, 2, (or more) of average edge length (v1, v2, v3). Edges are constructed to all nodes in the vicinity. After the current "cheapest" node is found, it becomes the standpoint node and is being surrounded by its vicinity. Certainly, the solutions of 3-vicinity lead to "straighter" trajectories, but it takes more complexity to compute them.

Superimposing Multiple Search Results

A better way of finding the generalized result of searching in the randomized graph is running search in multiple random grids and superimposing their solution. In Figure 6, we show the results of such superimposing of several min-cost paths for a dynamic system of

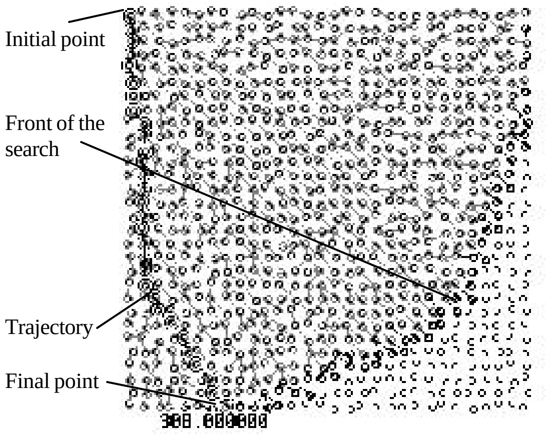
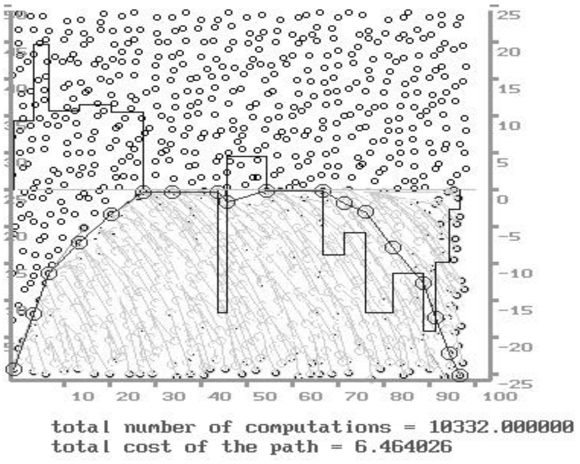


Figure 6: Velocity vs Distance with Friction (v1)

the first order (the acceleration and deceleration of a constant mass on a surface with friction. One can see that acceleration is not as rapid as deceleration. (This test further develops our prior results from [3]). It is important that in a multiplicity of superimposed random search results all stochastic components compensate for each other, and the envelope (the stripe for the subsequent higher resolution search) has smooth edges (as seen in Figure 9).

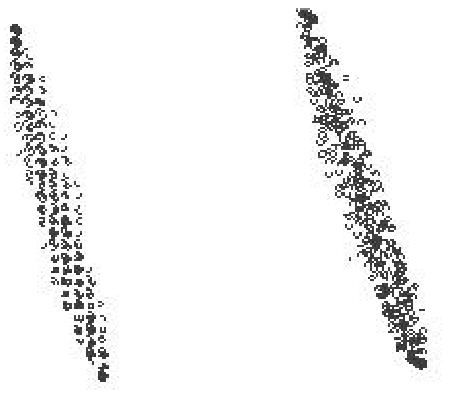


```
total number of computations = 10332.000000
total cost of the path = 6.464026
```

Figure 7: A single search performed on a grid with a shift of 30% and vicinity 1.

Solution to the grid problem

To avoid getting the extreme paths as the solutions we have to make sure that there aren't any "equidistant paths" from the starting to the destination point. This can be achieved by increasing the shift in the points there by eliminating the distorting effect of the grid.



(a) (b)

Figure 8. (a) Super imposed image of 30 runs of 30% shift (b) Super imposed image of 30 runs of 60% shift. The above runs were for performing a search of optimal path from the (0,0) location in the grid to (29,7) location

A comparison of the search with shift of 30% and 60% is shown in Figure 8 as the results of searching optimal path from the (0,0) location in the grid to (29,7) location It is clear from the Figure 8, the phenomenon of Bias is due to the grid effect. It was found that 50% shift is the most beneficial for removing the distorting grid effect. In other words, by shifting from a regular tessellation to a irregular tessellation we can reduce the bias problem substantially. Large bias is shown in Figure 6, a reduced case in Figure 8.

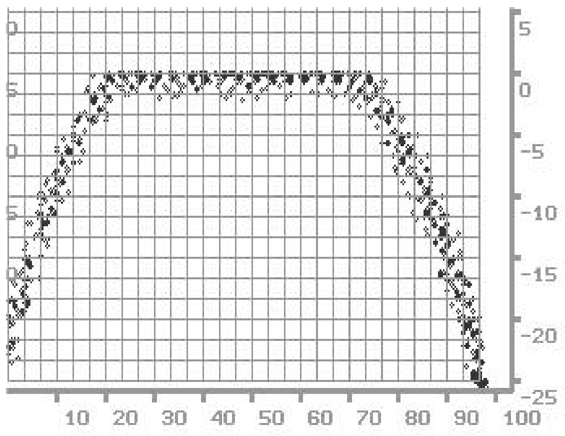


Figure 9. Envelope ("stripe") for the subsequent search of higher resolution obtained by superimposing multiple results of randomized searching.

Using randomization of the grid for virtual modeling the uncertainties

While using the tessellation for any particular purpose, 3 components affect actively the results of the execution.

1. *Grid Law:* The law is determined by the technique that we use to generate the tessellation.
2. *Grid Density:* This is computed as a ratio: the quantity of points located in the state space over its volume (area).
3. *Shift of the randomization:* This is the Results of "shaking" the grid (percentage) introduced to generate the effect of randomization.

Among these 3 components, the grid density and the shift of randomization create an intrinsic error of the path. When solving the same problem analytically, the solution (path) obtained has a stochastic component in it, which is perceived as a noise, a part of the uncertainties of the sources. This stochastic component is equivalent to the intrinsic error. Hence, if we know the sources of the uncertainty in the system (e. g. in the form of the values of Expectation and the Variance of this uncertainty) we can assign the grid density and the shift and the results of randomization will be equivalent to using the models of stochastic components, since we obtain the same Expectation and the Variance as the model would produce. In other words, instead of modeling the uncertainties we are can build the tessellation in such a manner that is produces this statistical truth.

Randomization in the Case of Multiple Traversabilities Segments

It is instructive to consider a case of finding the minimum cost path for a case of having multiple traversabilities of the state space. In Figure 10, a space segmented in multiple zones of traversbility is shown and the results of v1-searching for the case when traversability gradually reduces clockwise.

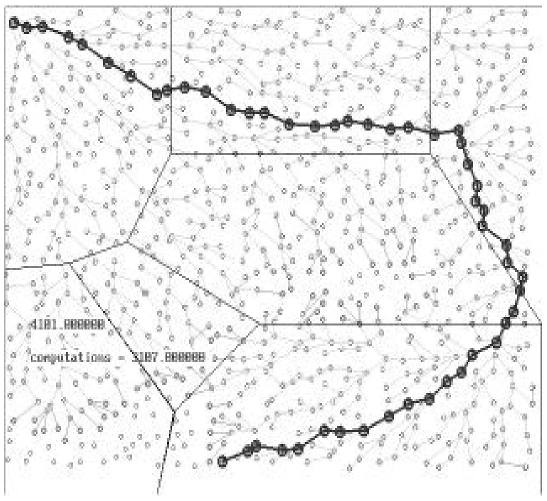


Figure 10. A single run of the S^3-search algorithm and a randomized grid

A question that should be answered: how to determine the width of a stripe (envelope) within which the search should be executed at a higher level of resolution. We exercised computing the boundaries for a stripe with 3σ width on the right and on the left around the average approximation of the path trajectory. This is a pretty cumbersome computation that includes the following stages: a) finding the approximation of the single path known, b) assuming that this approximation can be considered "an average recommended trajectory," c) computing the value of 3σ from the information of uncertainty, d) constructing piecewise boundaries around the average trajectory.

The following technique was tested and seems to be more practical. The search is being run many times, and the results of this multiple search are collected together and are considered a "stripe" (envelope).

In Figure 11, the results of this experiment are brought together for the case shown in Figure 10.

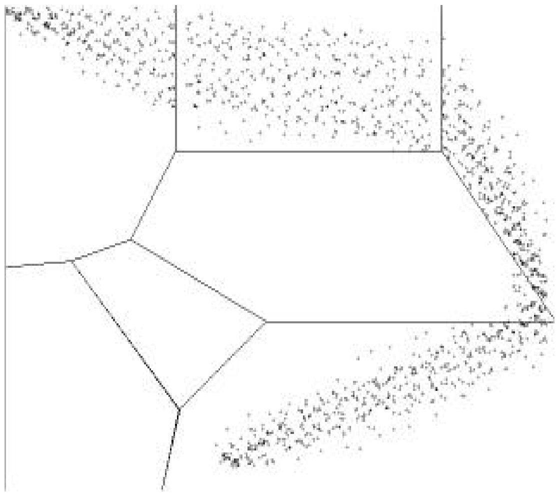


Figure 11. The cumulative results of conducting multiple search for the case shown in Figure 10.

One can see that a stripe emerges that demonstrate the statistical representation of the zone that is preferential for the subsequent higher resolution searching.

Interesting observations can be made: a) the stripe is narrowing as the traversability of tthe path is reducing, b) the stripe is narrowing in the areas of leaving IP and reaching the GP, c) in some areas the distribution gravitates to the unimodal law (like in Figure 8b); however in many areas the distribution is similar to the uniform one

References:

[1] A. Meystel, J. Albus. *Intelligent Systems: Architectures, Design, Control*, Wiley, NY, 2002, Pages 381-387

[2] A. Meystel, "Learning-Planning-Control Continuum," Ed. by H. S. Sarjoughian, et al, *2000 AI, Simulation and Planning in High Autonomy Systems*, SCS, San Diego, CA, 2000, pp. 3-23

3. A. Meystel, A. Lacaze, "Introduction to Integrated Planning-Learning Paradigm", in Proceedings of the 1997 International Conference on Intelligent Systems and Semiotics, Gaithersburg, MD, 1997, pp. 522-530